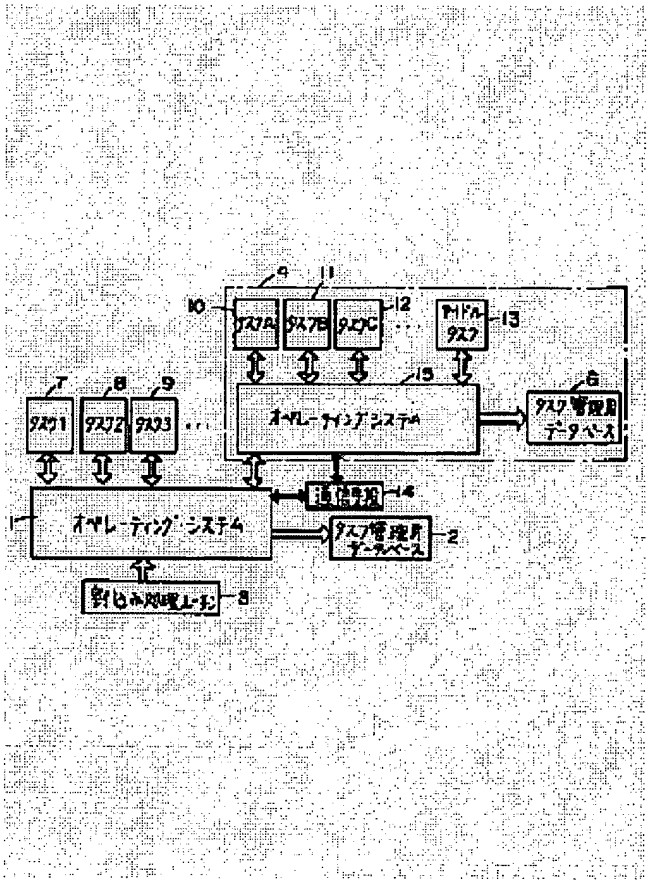
DATA PROCESSING SYSTEM

Patent number:	JP5108380
Publication date:	1993-04-30
Inventor:	SATO KOICHI
Applicant:	MITSUBISHI ELECTRIC CORP
Classification:	
- international:	G06F9/46
- european:	
Application number:	JP19910272846 19911021
Priority number(s):	

Abstract of **JP5108380**

PURPOSE:To provide the data processing system without deteriorating high-speed responsiveness even when the function of the system is more improved by providing independently first and second execution control data management means on the first and second execution control means. CONSTITUTION:The peripheral system of an OS(operating system) 15 operates as an idle task 4 under the management of the OS 1. The execution is shifted to the OS 15 when tasks 7 to 9 under the management of the OS 1 are finished, and the tasks 10 to 13 managed by the OS 15 are executed. When tasks 10 to 13 issue a system call, the OS 15 executes the processing. When a system call is generated, the context of the task under execution is saved in a database 6 for task management. The task scheduling is performed by the OS 15, and the data of the database 6 is operated. The context of a newly scheduled task is restored in a register group of hardware for task execution from the database 6.



(2)

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平5－108380

(43)公開日　平成５年(1993)４月30日

(51)Int.Cl.[5]	識別記号	庁内整理番号	Ｆ Ｉ	技術表示箇所
Ｇ０６Ｆ　9/46	３４０　Ｂ	8120－5B		
	Ｆ	8120－5B		

審査請求　未請求　請求項の数１(全　9　頁)

(21)出願番号　特願平3－272846

(22)出願日　平成３年(1991)10月21日

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号

(72)発明者　佐藤　浩一
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社エル・エス・アイ研究所内

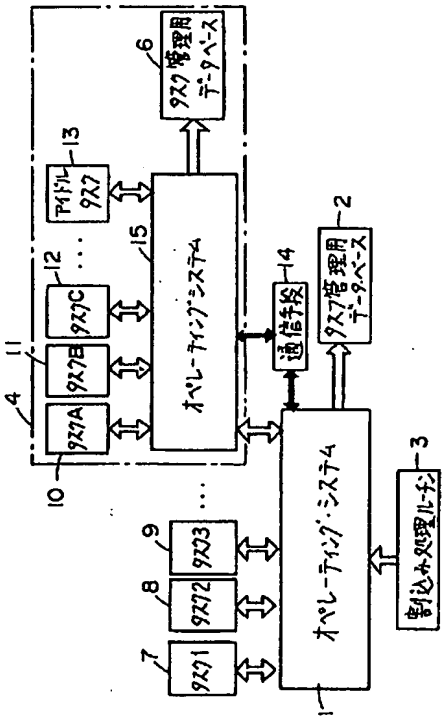
(74)代理人　弁理士　深見　久郎　（外３名）

(54)【発明の名称】　データ処理システム

(57)【要約】

【目的】　システムの機能がより高度化しても、高速応答性が低下しないデータ処理システムを提供する。

【構成】　タスク４、７～９を制御するＯＳ１と、ＯＳ１のタスク管理用のデータベース２と、ＯＳ１にタスクの１つとして割付けられ、タスク１０～１３の実行制御を行なうためのＯＳ５と、ＯＳ５のタスク管理のためのデータベース６とを含む。タスクを、そのコンテキストの大きさに応じて小さなものはＯＳ１に、大きなものはＯＳ５に割振ることが可能となる。



1

【特許請求の範囲】

【請求項１】　データ処理に必要なハードウェア資源およびソフトウェア資源と、

複数個の第１の仕事単位の各々に、所定の基準に従って前記ハードウェア資源およびソフトウェア資源を割当てることにより、各前記第１の仕事単位の実行状態を制御するための第１の実行制御手段と、

前記第１の実行制御手段による前記第１の仕事単位の実行状態の制御に必要なデータを格納して管理するための第１の実行制御データ管理手段と、

前記第１の実行制御手段に、前記第１の仕事単位の１つとして割付けられ、複数個の第２の仕事単位の各々に、所定の基準に従って前記ハードウェア資源およびソフトウェア資源を割当てることにより各前記第２の仕事単位の実行状態を制御するための第２の実行制御手段と、

前記第２の実行制御手段による前記第２の仕事単位の実行状態の制御に必要なデータを格納して管理するための第２の実行制御データ管理手段とを含むデータ処理システム。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】この発明は主としてコンピュータなどからなるデータ処理システムに関し、特に、高速応答性が要求されるリアルタイム処理のためのデータ処理システムに関する。

【０００２】

【従来の技術】従来のデータ処理システムの多くは、リアルタイム処理機能を備えたオペレーティングシステム（以後「ＯＳ」と省略する）により動作している。ＯＳは、コンピュータシステムの運用効率と操作性を高めるために、コンピュータシステムに組込まれたシステムプログラムを主として意味する。

【０００３】図６は、従来のデータ処理システムのソフトウェア構成の概念図である。図６を参照して、このデータ処理システムはソフトウェアとして、リアルタイムＯＳ６０と、ＯＳ６０の制御下で動作するタスク６３～６６と、これらタスク群６３～６６の実行を管理するためにＯＳ６０が使用するタスク管理用データベース６１と、オペレーティングシステム６０に対する割込の発生によって実行される割込処理ルーチン６２とを含む。

【０００４】各タスク６３～６６は、互いに独立した応用プログラムの最小仕事単位であり、必要に応じて個々に生成される。ＯＳ６０はこれらのタスク６３～６６の実行制御を管理し、いわゆるマルチタスキング処理を行なう。各タスク６３～６６には予め優先順位が与えられている。リアルタイムＯＳ６０は、各タスクに与えられた優先度に応じて、実行可能なタスクのうち最も高い優先度を有するタスクに実行権を与える。

【０００５】たとえば、現在実行中のタスクがＯＳの提供する機能を利用しようとする場合を考える。この場合

2

実行中のタスクは、システムコールと呼ばれる、システムに対する要求を発行する。実行中のタスクはシステムコールを発行した後、一旦待ち状態となる。実行中のタスクが待ち状態となれば、ＣＰＵが空き状態となる。資源を有効に利用するためには、この間にＣＰＵを他のタスクに割当ててこのタスクを実行させればよい。このとき、実行可能な複数のタスクがある場合、最も優先度の高いタスクが選択され実行される。また、現在実行中のタスクより優先度の高いタスクが実行可能状態になったとき（待ち状態が解除されたとき）は、現在実行中のタスクが一旦待ち状態にされ、その優先度の高いタスクに実行が移される。

【０００６】各タスクの状態はシステムコールにより遷移する。システムコールは実行中のタスクまたは割込処理ルーチンにより発行される。図７はその一例を示す状態遷移図である。図７を参照して、タスク１は優先度の低いタスク、タスク２は優先度の高いタスクであるものとする。時刻ｔ１以前にはタスク１が実行状態であり、タスク２が待ち状態になっているものとする。時刻ｔ１でタスク１がタスク２の待ち状態を解除するためのシステムコールを発行したとする。するとこの時点で実行は一時ＯＳに移る。ＯＳはタスク１からのシステムコールを受けてタスク２を実行可能状態とする。さらにＯＳは、タスク１とタスク２との優先度を比較して、次にどちらのタスクを実行すべきか判断する。どのタスクを実行すべきが判断することをタスクのスケジューリングと呼ぶ。その結果、優先度の高いタスクが時刻ｔ２に実行開始される。

【０００７】ＯＳがタスクのスケジューリングを行なう場合、タスクの実行状態や優先度などを表わす情報を記憶する必要がある。そのためにＯＳは、図６に示されるタスク管理用データベース６１を使用する。さらにＯＳは、複数のタスク間の通信や同期を実現するために、フラグやセマフォと呼ばれる様々なマルチタスキングのための機能を提供する。これらの機能を実現するためには、システムに共通のデータベースが必要とされる。このデータベースは、タスク管理用データベース６１に含まれる。ＯＳが各タスクに対して提供する機能が複雑になるほどタスク管理用データベース６１に格納すべき情報は増加する。また、それにつれて１回のシステムコールの発生によって操作されるデータ量も増加する。

【０００８】図７において、ＯＳによる処理の前後でタスク１からタスク２に実行が移っている。このことを「タスクをディスパッチする」と呼ぶ。タスクのディスパッチは、ＯＳにより以下のように行なわれる。まず、ディスパッチ前のタスクが使用していたハードウェアのレジスタの内容を図６のタスク管理用データベース６１にセーブする。次に、ディスパッチ後に実行されるタスクのレジスタ内容を、タスク管理用データベース６１からタスクを実行するためのハードウェアのレジスタにリ

ストアする。タスク管理用データベース６１には、このディスパッチ後に実行されるタスクが前回処理中断したときのハードウェアのレジスタ内容が格納されており、リストアにより前回の処理中断時の状態にハードウェアの各レジスタ内容が戻る。セーブ・リストアされるハードウェアのレジスタとしては、ＭＰＵ（マイクロプロセッサユニット）の制御レジスタや汎用レジスタ、ＦＰＵ（浮動小数点演算ユニット）の浮動小数点レジスタなどがある。これらハードウェアのレジスタ内容をタスクのコンテキストと呼ぶ。

【０００９】システムコールは実行中のタスクのほかに、割込処理ルーチンからも発行される。この場合もシステムコールが発行されると一時実行はＯＳに移る。ＯＳはこのシステムコールに応答して対応する処理を行なう。

【００１０】ところで、ＯＳがシステムコールを処理しているときに割込を無制限に受付けると、割込処理の中でタスク管理用データベース６１の内容が操作され、データベースの一意性が失われる可能性がある。このような事態となればタスク管理は不可能となる。これを防ぐために、通常、ＯＳが上述のシステムコールに対する処理を行なっているような場合には、割込は禁止される。ただし、リアルタイムＯＳは高速応答性を備えていることが絶対条件である。特に割込禁止時間が長くなると、外部要因に対するシステムの応答性能に直接悪影響を及ぼし好ましくない。そのため、リアルタイムＯＳの設計時には、割込禁止時間が可能な限り短くなるよう最大の注意が払われる。

【００１１】図８は、システムコールが発行されたときのＯＳの処理の一例を示すフローチャートである。システムコール入口からこの処理に入った場合、ステップＳ８１でまず割込が禁止される。続いてステップＳ８２で、実行していたタスクのコンテキストをデータベース（タスク管理用データベース６１）にセーブする。さらにステップＳ８３で、データベース６１を操作してタスクのスケジューリングが行なわれる。ステップＳ８４で、スケジューリングの結果次に実行されるタスクとして選択されたタスクのコンテキストをタスク管理用データベース６１からタスク実行用のハードウェアのレジスタ群にリストアする。さらにステップＳ８５で割込禁止が解除される。そしてシステムコールの処理は終了する。

【００１２】図６において、ＯＳにより実行制御されるタスク６６は、「アイドルタスク」と呼ばれるものである。アイドルタスク６６は、システムにおいて最も優先度の低いタスクである。このアイドルタスクを常にシステム内に存在させておくことにより、このシステム内において実行されるべきタスクが１つもないという状態があり得なくなる。

【００１３】

【発明が解決しようとする課題】従来のデータ処理システムで、システムの高機能化に伴いＯＳが複雑な機能を提供する必要が増大している。この場合ＯＳが操作するデータベースは大きくなり、その結果一度のシステムコールを処理するための時間は長くなり、コンテキストのセーブ・リストア時にオーバーヘッドが生ずる。マルチタスキング処理においてタスクの実行以外に必要な時間が増大することとなり、高速応答性を備えるべきリアルタイムＯＳとしては好ましくない。また、割込禁止時間も長くなり、割込に対する時間的な応答性が劣化する。

【００１４】それゆえにこの発明の目的は、システムの機能がより高度化しても、高速応答性が低下しないデータ処理システムを提供することである。

【００１５】

【課題を解決するための手段】本発明に係るデータ処理システムは、データ処理に必要なハードウェア資源およびソフトウェア資源と、複数個の第１の仕事単位の各々に、所定の基準に従ってハードウェア資源およびソフトウェア資源を割当てることにより、各第１の仕事単位の実行状態を制御するための第１の実行制御手段と、第１の実行制御手段による第１の仕事単位の実行状態の制御に必要なデータを格納して管理するための第１の実行制御データ管理手段と、第１の実行制御手段に、第１の仕事単位の１つとして割付けられ、複数個の第２の仕事単位の各々に、所定の基準に従ってハードウェア資源およびソフトウェア資源を割当てることにより、各第２の仕事単位の実行状態を制御するための第２の実行制御手段と、第２の実行制御手段による第２の仕事単位の実行状態の制御に必要なデータを格納して管理するための第２の実行制御データ管理手段とを含む。

【００１６】

【作用】上述のデータ処理システムにおいては、第１の仕事単位と第２の仕事単位とは、それぞれ別個の第１の実行制御手段および第２の実行制御手段によって実行制御される。各実行制御手段の提供する機能を異ならせ、あまり多くの機能を要求しないタスクを実行制御するための実行制御手段には比較的少ない機能を、より多数の機能を提供する必要のある実行制御手段には比較的多数の機能をそれぞれ備えるように各実行制御手段の内容を異ならせることができる。また、第１の実行制御手段および第２の実行制御手段の各々に別個の実行制御データ管理手段を設けたため、各実行制御手段におけるタスク管理の際のコンテキストの内容を異ならせることができる。したがって、特定のタスクを、そのタスクがシステムに対して要求する機能やタスク管理のために必要なコンテキストの内容に応じて最も適切な実行制御手段の管理の下で実行させることができる。

【００１７】

【実施例】図２は、この発明の一実施例に係るデータ処理システムのハードウェア構成図である。図２を参照し

て、このデータ処理システム２７は、プログラムを実行するためのＭＰＵ２０と、ＭＰＵ２０に接続され、データ処理システム２７に含まれる各構成要素間でデータをやりとりしたり、ＭＰＵ２０に対して割込信号を入力したりするための内部バス２６と、内部バス２６に接続され、浮動小数点演算を行なうためのＦＰＵ２８と、本システムを制御したり、データ処理を行なったりするためのプログラムおよびシステムを管理するためのデータベースと、処理すべきデータなどとを格納するためのメモリ２１と、内部バス２６に接続され、ＣＲＴ２９を制御するためのＣＲＴコントローラ２２と、内部バス２６および外部のネットワーク３０に接続され、通信制御を行なうための通信コントローラ２３と、内部バス２６に接続され、ディスクユニット３１などの外部記憶装置の制御を行なうためのディスクコントローラ２４と、内部バス２６に接続され、各種センサ３２からの信号を変換してＭＰＵ２０に与えるためのセンサコントローラ２５とを含む。

【００１８】図２に示される各構成要素のうち、各コントローラ２２～２５は、たとえばデータの入出力が完了したり、外部で変化が生じたことを検知したときに、割込信号を内部バス２６を介してＭＰＵ２０に与える。また、データ処理に必要なソフトウェア資源およびタスク管理のためのオペレーティングシステムなどのシステム資源もメモリ２１に格納されている。

【００１９】図１は、この発明の一実施例に係るデータ処理システムを形成するソフトウェア資源の構成図である。このソフトウェア資源は、前述のように図２に示されるメモリ２１に格納されている。図１を参照して、このシステムは、制限された機能のみを各タスクに対して提供するリアルタイムＯＳ１と、ＯＳ１の管理下で動作するタスク７～９と同じくＯＳ１の管理下で動作するアイドルタスク４と、ＯＳ１がタスク４、７～９の実行制御を管理するために用いるためのタスク管理用データベース２と、ＯＳ１の管理下で処理される割込処理ルーチン３とを含む。

【００２０】アイドルタスク４は１つのシステムを構成しており、ＯＳ１のアイドルタスクとして実行される、ＯＳ１よりも複雑な機能を提供するリアルタイムＯＳ５と、ＯＳ５によって管理され実行される複数個のタスク１２～１４と、ＯＳ５によって管理されるアイドルタスク１３と、ＯＳ５がタスク１０～１３の実行制御を管理するために必要なデータを格納するためのタスク管理用データベース６とを含む。ＯＳ５は、ＯＳ１と異なり割込処理機能は持たない。

【００２１】このシステムはさらに、ＯＳ１とＯＳ２との間に設けられ、ＯＳ１とＯＳ５との間で、たとえば特定の処理の実行の依頼などを通信するための通信手段１４を含む。

【００２２】図１に示されるデータ処理システムは以下のように動作する。ＯＳ１は、図６に示されるリアルタイムＯＳ６０と同様に動作する。すなわち、ＯＳ１は、その管理するタスク７～９がシステムコールを発行すると、ＯＳ１に対する割込を禁止してから、このシステムコールの処理を行なう。このとき、前述のようにＯＳ１が提供する機能は制限されたものである。そのため、システムコール処理時にデータベース２を操作する量は、提供する機能が複雑な場合と比べて少なくすることができる。したがってＯＳ１がシステムコールを処理するために要する時間は短くなる。前述のようにシステムコール処理時間には、割込が禁止される。したがってシステムコール処理時間が短くなることによりＯＳ１に対する割込禁止時間も短くでき、割込に対する応答性が向上する。

【００２３】図１に示されるシステムが従来のシステムと異なるのは、ＯＳ５ををを中心とするシステムがアイドルタスク４としてＯＳ１の管理下で動作することである。ＯＳ１の管理下で動作するタスク７～９がなくなったときにＯＳ５に実行が移り、ＯＳ５が管理するタスク１０～１３が実行される。タスク１０～１３がシステムコールを発行したときは、ＯＳ５がその処理を行なう。

【００２４】ＯＳ５で行なわれるシステムコール処理のフローチャートを図３に示す。図３を参照して、システムコールが発生すると、ステップＳ３１において実行中のタスクのコンテキストがタスク管理用データベース６にセーブされる。ステップＳ３２において、ＯＳ５によってタスクのスケジューリングが行なわれる。このとき、タスク管理用データベース６のデータが操作される。続いてステップＳ３３で、スケジューリングされた新タスクのコンテキストがタスク管理用データベース６からタスク実行用のハードウェアのレジスタ群にリストアされ、システムコール処理が終了する。

【００２５】図３と図８とを対比してすぐわかるように、ＯＳ５によるシステムコール処理においては、システムコール実行中でも割込が禁止されない。これは、本発明の実施例に係るデータ処理システムにおいては、図１に示されるように、割込処理ルーチン３がＯＳ１の管理下でのみ行なわれ、ＯＳ５においては割込処理を行なう必要がないためである。割込処理によってＯＳ５のタスク管理用データベース６が操作されることはない。したがってＯＳ５システムコール処理においては、割込禁止とする必要はない。ＯＳ５の機能をどれだけ複雑にしても、ＯＳ５のシステムコール処理実行中に割込禁止とする必要はないため、このデータ処理システム全体の割込に対する応答性を劣化させずにすむ。それに対してＯＳ１においては割込処理を行なう必要がある。そのため、ＯＳ１は、制限された機能のみタスクに提供する。これによりシステムコール処理時にＯＳ１がデータベース２を操作する量は少なくなり、ＯＳ１のシステムコール処理時間が短くなる。割込禁止時間も短くすることが

でき、割込に対する応答性を高めることができる。

【００２６】さらに、浮動小数点計算など、大きなコンテキストを持つタスクはＯＳ５で、小さなコンテキストで実行可能なタスクはＯＳ１の管理下で、それぞれ実行させることができる。これにより、たとえば小さなコンテキストを有するタスクについてコンテキトのセーブ・リストアを行なう場合、従来よりも小さなコンテキストのセーブ・リストアのみでシステムコール処理を行なうことができる。大きなコンテキストのタスクについては、そのコンテキストのセーブ・リストアはＯＳ５で行なわれる。したがって、従来のように各タスクのコンテキストの大小にかかわらず最も大きなコンテキストに合わせたセーブ・リストアを行なうような無駄を省くことができ、ＯＳ１の管理下のタスクからシステムコールを処理するための時間を短縮することができる。

【００２７】一方、場合によっては、図１に示されるＯＳ１の管理下で実行されるタスクのいずれかから、システムコールを発行してＯＳ５に属するタスクを操作したい場合もあると考えられる。通信手段１４はこれを可能とするために設けられたものである。ＯＳ１に属するタスクが発行したシステムコールの処理は通信手段１４を介してＯＳ１からＯＳ５に依頼される。そしてＯＳ５がこのシステムコールに対する処理を行なう。

【００２８】この第１の実施例のデータ処理システムにおいては、ＯＳ１のアイドルタスク４としてＯＳ５を割付けている。このようにすることにより、ＯＳ１に属するタスク７～９には、ＯＳ５に属するタスク１０～１２よりも高い優先度が自動的に与えられることになる。これは、以下のような理由による。ＯＳ１は、高速にシステムコールを処理するためにその提供する機能を少なくし、また各タスクのコンテキストを軽くしている。つまり、高速に実行したいタスクはＯＳ１の管理下で実行させることが前提とされている。このような条件の下では、ＯＳ１に属するタスクは、当然ＯＳ５に属するタスクよりも高い優先度を有するべきである。

【００２９】図４は、この発明の第２の実施例に係るデータ処理システムのソフトウェア構成図である。図４に示されるデータ処理システムが図１に示されるデータ処理システムと異なるのは、第２のＯＳ５を中心とするシステムが、第１のＯＳ１の、アイドルタスク以外のタスク４に割付けられることである。その他の点において、図４に示されるシステムは図１に示されるシステムと同様である。同一の構成要素には同一の参照番号および名称が与えられている。それらの機能も同一である。したがって、ここではそれらについての詳しい説明は繰り返されない。

【００３０】図４に示されるシステムにおいては、第２のＯＳ５に属するタスクの優先度が自動的に第１のＯＳ１に属するタスクの優先度よりも低くなるということはない。この優先度はＯＳ１に割付けられる各タスクの優先度と、ＯＳ５に従属する各タスクに割当てられる優先度とから決まる。このようにすることによっても、ＯＳ１による割込処理に対する高速応答が実現でき、またＯＳ１に従属するタスクとしてはコンテキストの小さなものを選択することにより、システムのオーバーヘッドが増加するおそれがなく、リアルタイムＯＳとして望ましい高速応答性を備えることができる。

【００３１】図５には、本発明に係るデータ処理システムの第３の実施例のソフトウェア構成図が示されている。図５に示されるデータ処理システムが図１に示されるデータ処理システムと異なるのは、アイドルタスク４内のシステムのＯＳ５に従属するタスクのうちの１つが、第３のＯＳ５１を含むことと、通信手段１４がＯＳ１、５に加えてＯＳ５１とも通信可能なこととである。

【００３２】図５を参照して、アイドルタスク１３は、ＯＳ５の管理下で動作するＯＳ５１と、ＯＳ５１の管理下で動作するタスク５３～５４と、ＯＳ５１がタスク５３～５４の実行制御を管理するために使用するタスク管理用データベース５２とを含む。

【００３３】タスク５４はＯＳ５１の管理下で動作するアイドルタスクである。ＯＳ５１を中心とするシステムとＯＳ５を中心とするシステムとの関係は、ＯＳ５を中心とするシステムとＯＳ１を中心とするシステムとの関係と同様である。このシステムでは優先度はＯＳ１の従属タスク、ＯＳ５の従属タスク、ＯＳ５１の従属タスクという順になる。各ＯＳ１、５、５１には、順にコンテキストの小、中、大のタスクが割当てられる。このようにすることにより、各タスクのコンテキストの大小に応じてシステムコール処理時のデータのセーブ・リストア量を変更することができる。そのため、システムコール処理時のシステムのオーバーヘッドが減少するとともに、ＯＳ１のみで行なわれる割込処理に対する応答性も良好なものに保たれる。また、４つ以上のＯＳを使用して図５に示されるようなシステムを構成することもできる。

【００３４】以上のようにこの発明に係るデータ処理システムにおいては、互いに独立したタスク管理用のデータベースを備えた複数のＯＳが使用される。割込処理はそのうちの１つのＯＳでのみ行なわれる。他のＯＳでは割込処理が行なわれない。そのため他のＯＳのシステムコール処理が複雑になっても、割込応答性を劣化させずにすむ。また、各タスクをそのコンテキストの大きさに応じて適切なＯＳの管理下で動作させることができる。したがって冗長なコンテキストのセーブ・リストアを省くことができ、システムのオーバーヘッドを取除くことができる。

【００３５】

【発明の効果】以上のようにこの発明に係るデータ処理システムによれば、第１および第２の実行制御手段に、それぞれ独立の第１および第２の実行制御データ管理手

段を設けた。したがって各仕事単位を、その実行制御に必要なコンテキストの大きさに従って第１または第２の実行制御手段のうちの適切なものの管理下で動作させるように割振ることができる。冗長なコンテキストのセーブ・リストアを省くことが可能となり、システムの動作をより効率よくでき、その応答性も良好にすることができる。また、第１の実行制御手段と第２の実行制御手段とによって提供される機能を、その複雑さにおいて異なったものとすることができる。したがって第１の実行制御手段より提供される機能をより簡単なものとすることにより、第１の実行制御手段の管理下の第１の仕事単位についての処理を高速化できる。また、第２の実行制御手段により提供される機能をより複雑なものとすることにより、より多くの機能を実行制御手段に要求するような第２の仕事単位も実行することが可能である。

【００３６】その結果、システムの機能がより高度化しても、高速応答性が低下しないデータ処理システムを提供することができる。

【図面の簡単な説明】

【図１】図１は、本発明の第１の実施例に係るデータ処理システムのソフトウェア構成図である。

【図２】図２は、本発明の第１の実施例に係るデータ処理システムのハードウェア構成図である。

【図３】図３は、第１の実施例の、第２のＯＳで行なわ＊れるシステムコール処理のフローチャートである。

【図４】図４は、本発明の第２の実施例に係るデータ処理システムのソフトウェア構成図である。

【図５】図５は、本発明の第３の実施例に係るデータ処理システムのソフトウェア構成図である。

【図６】図６は、従来のリアルタイムＯＳを用いたデータ処理システムのソフトウェア構成図である。

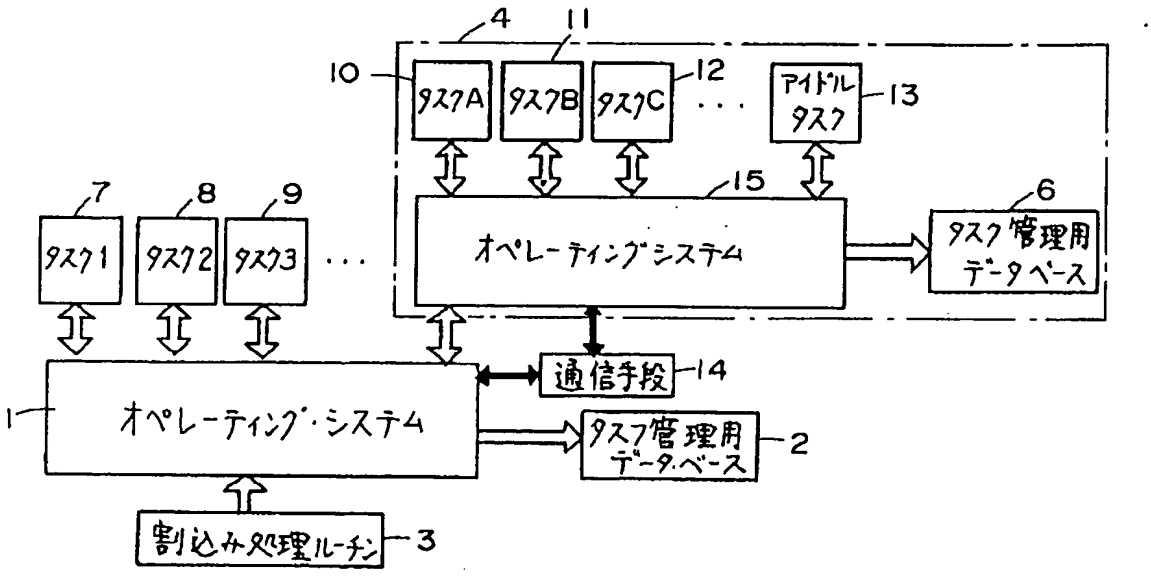
【図７】図７は、タスクのディスパッチの様子を示す模式図である。

【図８】図８は、従来のリアルタイムＯＳのシステムコール処理のフローチャートである。

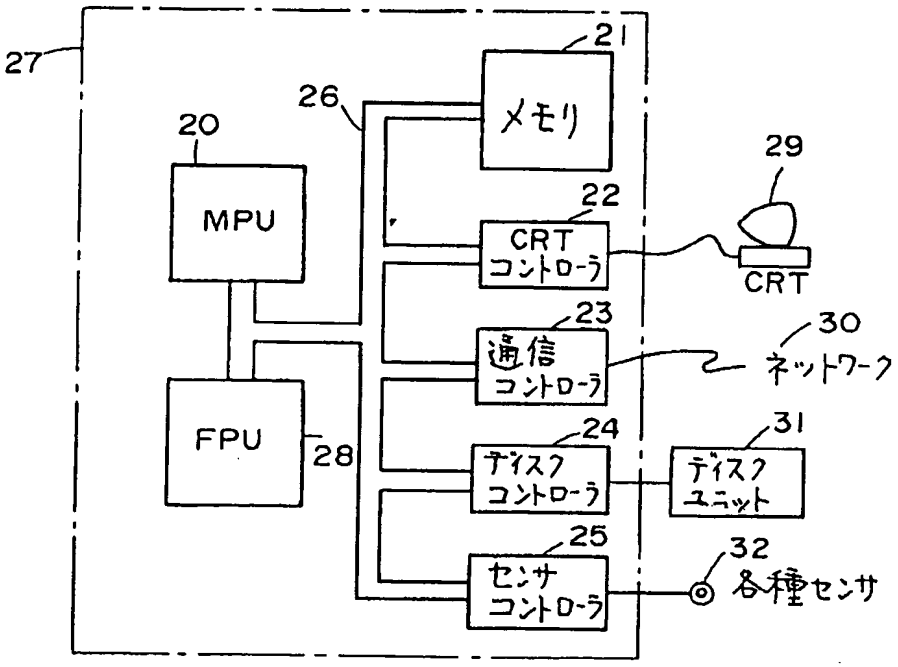
【符号の説明】

１　リアルタイムＯＳ
２　ＯＳ１が使用するデータベース
３　割込処理ルーチン
５　割込処理に関与しないＯＳ
６　ＯＳ５が使用するデータベース
７～９　ＯＳ１の管理下で動作するタスク群
１０～１２　ＯＳ５の管理下で動作するタスク群
１３　ＯＳ５のアイドルタスク
１４　ＯＳ１とＯＳ５の間の通信手段
２０　マイクロプロセッサユニット
２１　メモリ
２６　内部バス

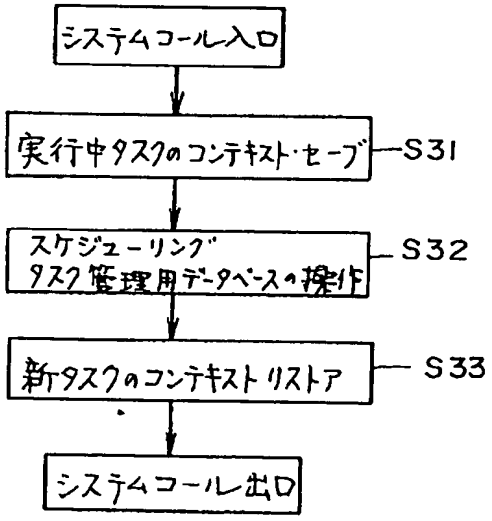
【図１】



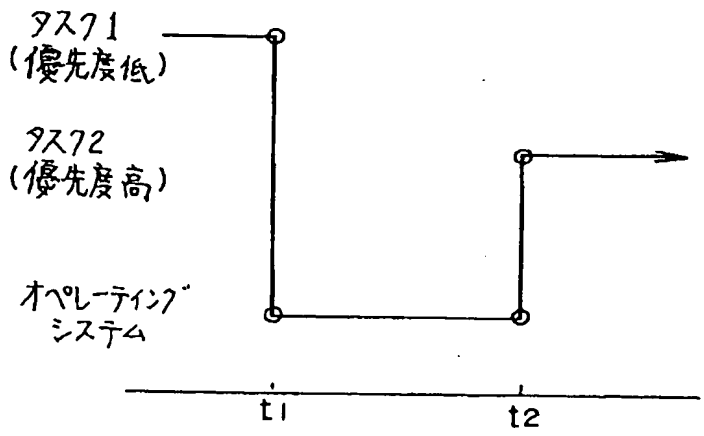
【図２】


27
21
メモリ
26
20
MPU
29
22
CRT
コントローラ
CRT
23
30
通信
コントローラ
ネットワーク
FPU
28
24
31
ディスク
コントローラ
ディスク
ユニット
25
32
センサ
コントローラ
各種センサ

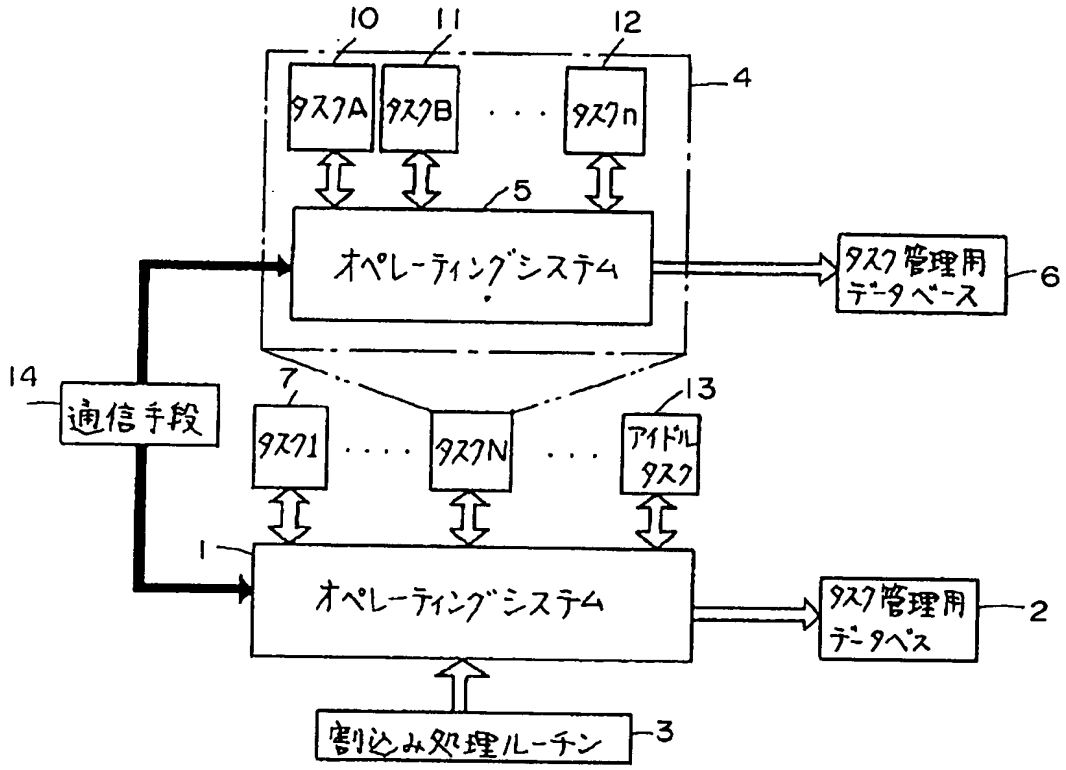
【図３】


システムコール入口
実行中タスクのコンテキスト・セーブ
S31
スケジューリング
タスク管理用データベースの操作
S32
新タスクのコンテキストリストア
S33
システムコール出口

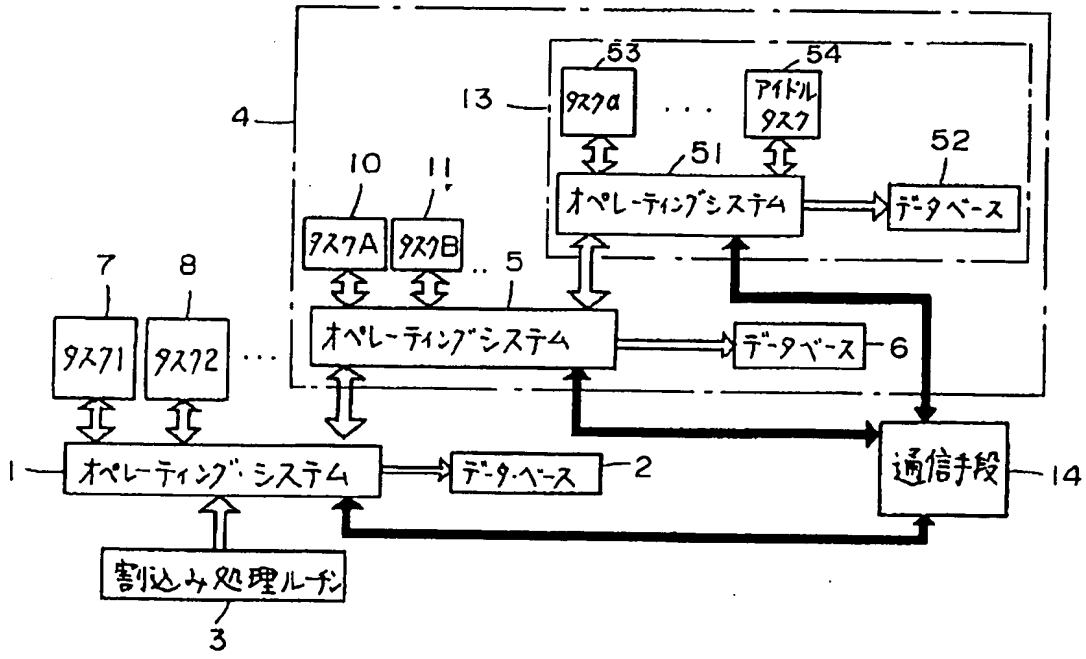
【図７】


タスク1
(優先度低)
タスク2
(優先度高)
オペレーティング
システム
t1
t2

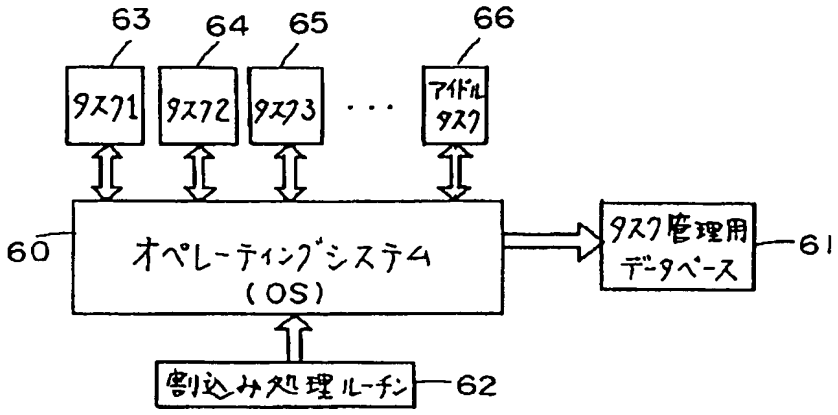
【図４】


10
11
12
タスクA
タスクB
タスクn
4
5
オペレーティングシステム
タスク管理用データベース
6
14
通信手段
7
タスク1
タスクN
13
アイドルタスク
1
オペレーティングシステム
タスク管理用データベース
2
割込み処理ルーチン
3

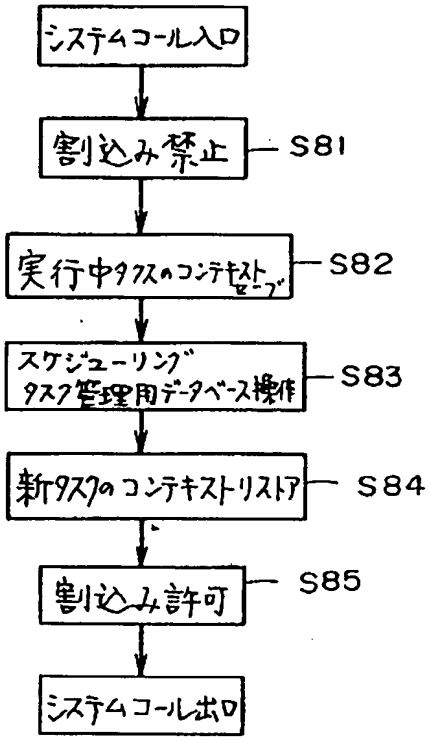
【図５】


4
13
53
タスクa
54
アイドルタスク
52
51
オペレーティングシステム
データベース
10
11
タスクA
タスクB
5
7
8
タスク1
タスク2
オペレーティングシステム
データベース
6
1
オペレーティングシステム
データベース
2
通信手段
14
割込み処理ルーチン
3

【図６】


63
64
65
66
タスク1
タスク2
タスク3
・・・
アイドルタスク
60
オペレーティングシステム
(OS)
タスク管理用データベース
61
割込み処理ルーチン
62

【図８】


システムコール入口
割込み禁止
S81
実行中タスクのコンテキストセーブ
S82
スケジューリング
タスク管理用データベース操作
S83
新タスクのコンテキストリストア
S84
割込み許可
S85
システムコール出口